Carnegie Mellon University
Software Engineering Institute

Product Line Practice Workshop Report

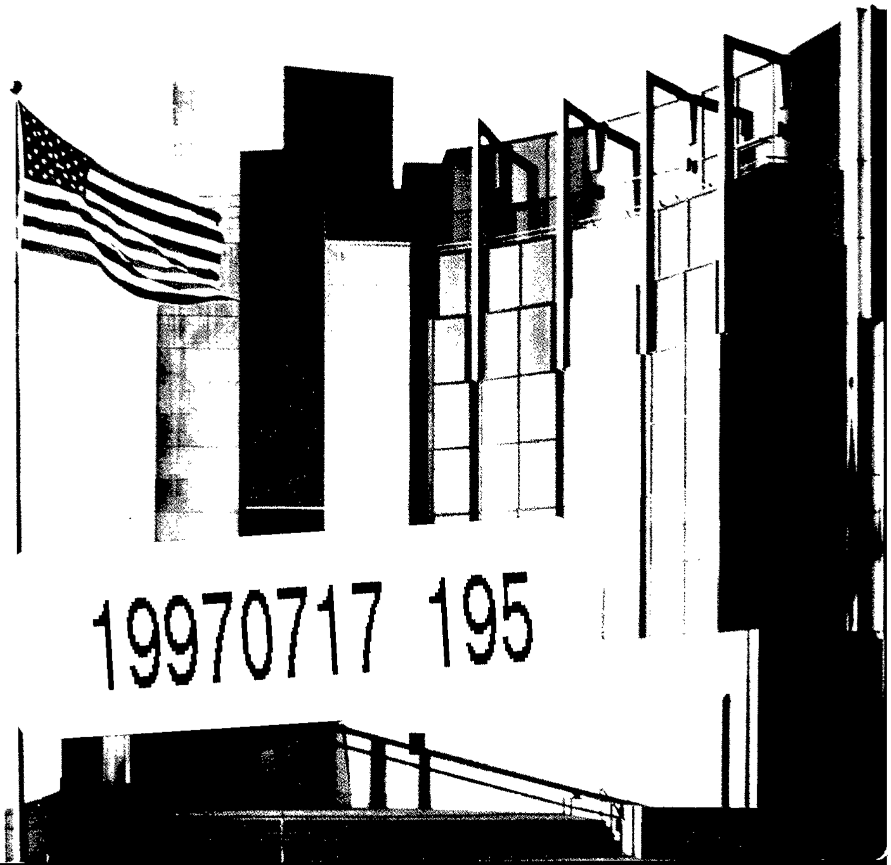
Len Bass
Paul Clements
Sholom Cohen
Linda Northrop
James Withey

June 1997

TR

TECHNICAL REPORT
CMU/SEI-97-TR-003
ESC-TR-97-003

DTIC QUALITY INSPECTED 3



19970717 195

Carnegie Mellon University does not discriminate and Carnegie Mellon University is required not to discriminate in admission, employment, or administration of its programs or activities on the basis of race, color, national origin, sex or handicap in violation of Title VI of the Civil Rights Act of 1964, Title IX of the Educational Amendments of 1972 and Section 504 of the Rehabilitation Act of 1973 or other federal, state, or local laws or executive orders.

In addition, Carnegie Mellon University does not discriminate in admission, employment or administration of its programs on the basis of religion, creed, ancestry, belief, age, veteran status, sexual orientation or in violation of federal, state, or local laws or executive orders. However, in the judgment of the Carnegie Mellon Human Relations Commission, the Department of Defense policy of, "Don't ask, don't tell, don't pursue," excludes openly gay, lesbian and bisexual students from receiving ROTC scholarships or serving in the military. Nevertheless, all ROTC classes at Carnegie Mellon University are available to all students.

Inquiries concerning application of these statements should be directed to the Provost, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, telephone (412) 268-6684 or the Vice President for Enrollment, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, telephone (412) 268-2056.

Obtain general information about Carnegie Mellon University by calling (412) 268-2000.

Technical Report
CMU/SEI-97-TR-003
ESC-TR-97-003
June 1997

Product Line Practice Workshop Report

Len Bass
Paul Clements
Sholom Cohen
Linda Northrop
James Withey

Product Lines Systems

DTIC QUALITY INSPECTED 3

Unlimited distribution subject to the copyright.

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

This report was prepared for the

SEI Joint Program Office
HQ ESC/AXS
5 Eglin Street
Hanscom AFB, MA 01731-2116

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

FOR THE COMMANDER

Thomas R. Miller

Thomas R. Miller, Lt Col, USAF
SEI Joint Program Office

This work is sponsored by the U.S. Department of Defense.

Copyright © 1997 by Carnegie Mellon University.

Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and "No Warranty" statements are included with all reproductions and derivative works.

Requests for permission to reproduce this document or to prepare derivative works of this document for external and commercial use should be addressed to the SEI Licensing Agent.

NO WARRANTY

THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

This work was created in the performance of Federal Government Contract Number F19628-95-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 52.227-7013.

This document is available through SAIC/ASSET: 1350 Earl L. Core Road; PO Box 3305; Morgantown, West Virginia 26505 / Phone: (304) 284-9000 / FAX: (304) 284-9001 / World Wide Web: http://www.saic.com/contact.html / e-mail: webmaster@cpqm.saic.com

Copies of this document are available through the National Technical Information Service (NTIS). For information on ordering, please contact NTIS directly: National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161. Phone: (703) 487-4600.

This document is also available through the Defense Technical Information Center (DTIC). DTIC provides access to and transfer of scientific and technical information for DoD personnel, DoD contractors and potential contractors, and other U.S. Government agency personnel and their contractors. To obtain a copy, please contact DTIC directly: Defense Technical Information Center / Attn: BRR / 8725 John J. Kingman Road / Suite 0944 / Ft. Belvoir, VA 22060-6218. Phone: (703) 767-8274 or toll-free in the U.S. — 1-800 225-3842).

Use of any trademarks in this report is not intended in any way to infringe on the rights of the trademark holder.

Table of Contents

List of Figures

List of Tables

Product Line Practice Workshop Report

Abstract: The first Software Engineering Institute Product Line Practice Workshop was a hands-on meeting held in December 1996 to share industry and government practices in software product lines and to explore the technical and non-technical issues involved. This report synthesizes the workshop presentations and discussions, which identified factors involved in product line practices and analyzed issues in the areas of architecture, people-organization-management, and business models.

1 Introduction

1.1 Why Product Line Practice?

Historically, software engineers have designed software systems one system at a time. Each software product involves significant investment in requirements analysis, architecture and design, documentation, prototyping, process and method definition, tools, training, implementation, and testing. Little attention has been paid to the consequences of a design in the production of multiple software-intensive products.

Some organizations now realize that they can no longer afford to develop multiple software products one product at a time. To retain market share they are pressured to introduce new products and add functionality to existing products at a rapid pace. These organizations have instead adopted a p*roduct line approach* that uses a common set of core assets to modify, assemble, instantiate, or generate multiple products referred to as a product line. Such a product line approach involves building a product line as a product family.

A *product line* is defined as a group of products sharing a common, managed set of features that satisfy specific needs of a selected market or mission. A group of systems built from a common set of assets is a *product family.* A product family need not constitute a product line; the individual systems may have no clearly coordinated role addressing a single market. Conversely, a product line need not be built from a product family; the individual systems in the product line may be developed independently. However, building a product line as a product family leverages and amortizes prior investment to the maximum degree possible. Assuming the product line has a sufficient number of members, the sum cost of building the product line as a whole is much less than the sum cost of building the individual systems independently. [1]

[1] Henceforth, when we speak of a product line we mean one that is developed using a common set of core assets, i.e., a product line built as a product family.

The software architecture is one of the important assets shared by the systems in a product line. A *software architecture* describes the structural properties of the software—typically the components and their interrelationships and guidelines about their use [Clements 96]. A software architecture that capitalizes on commonalities in the implementation of the line of products can provide the structural robustness that makes the derivation of software products from software assets economically viable.

Some organizations have already experienced considerable savings by using a product line approach for software system production. Other organizations are attracted to the idea but are in varying stages of using product line practices.

In January 1997, the Software Engineering Institute (SEI) launched a technical initiative, the Product Line Practice Initiative, to help facilitate and accelerate the transition to sound software engineering practices using a product line approach. The goal of the Product Line Practice Initiative is to provide organizations with an integrated business and technical approach to multi-use of software assets so that organizations can produce and maintain similar systems of predictable quality at lower cost. One of the strategies to reach this goal involves direct interaction with and nurturing of the community interested in product line practice. The Product Line Practice Workshop described in this report is just one of the SEI efforts to execute this strategy.

1.2 About the Workshop

In order to connect with the product line practice community and to learn the factors and issues in current industry and government approaches to software product lines, the SEI held a two-day Product Line Practice Workshop in December 1996. The participants in this workshop were invited based upon our knowledge of their company's experience with and/or commitment to software product lines and/or strategic software reuse. Together we formulated and discussed the issues that form the backbone of this report.

The workshop participants included

Felix Bachmann, Bosch USA/SEI Resident Affiliate
Len Bass, Product Line Systems Program, SEI
Ed Betts, Caterpillar, Inc.
Gary Chastek, Product Line Systems Program, SEI
Paul Clements, Product Line Systems Program, SEI
Sholom Cohen, Product Line Systems Program, SEI
Ed Gebauer, Industry Sector, SEI
James McLaughlin, Schlumberger Well Services
Linda Northrop, Manager, Product Line Systems Program, SEI
J. Henk Obbink, Philips Laboratories
Jaak Urmi, CelsiusTech Systems AB
John Willison, U.S. Army CECOM

2 Product Line Experiences: Summary of Participants' Presentations

Each guest at the workshop was asked to make a presentation explaining his organization's approach to developing software product lines. The following questions were provided to participants beforehand to provide a common structure for the presentations:

- What kinds of software product lines have you developed?
- What was the context (technical, organizational, problem domain) for creating the product lines?
- What techniques are you using/did you use?
- What were the costs?
- What were the benefits?
- What results (and measures) have been collected?
- How have you validated your results?
- What were the risks and how did you mitigate them?
- What are the open issues (technical and non-technical) that you feel need to be resolved relative to software product lines?

These questions established a common frame of reference that allowed the participants to communicate with each other despite their diverse backgrounds. We have already reported the kinds of software products produced by our participants in Section 1.2, and many of the remaining questions will be addressed in this section. However, the purpose of the workshop was not to catalog individual product line approaches[1] but rather to synthesize the combined experience of many organizations to begin to build a coherent model of product line practice. Therefore, this section summarizes the presentations as a group, rather than individually, by re-casting the presentations in terms of a set of common themes that arose (several of these were raised by the original interview questions). These themes are *contextual, technology, organizational,* and *business factors.* We will treat each one in turn.

2.1 Contextual Factors

Contextual factors describe the environment in which the organization exists or existed when it launched the product line effort; this includes a description of its goals, the technical assets in place, its business state, and how the organization is/was situated in its marketplace.

[1] Other work in the Product Line Systems Initiative at the SEI will capture and record illuminating product line case studies. See for example [Brownsword 96].

Key specific areas include:

- **Product line assets designed or mined.** A product line can be designed from the ground up or can be built using assets from previous efforts. A designed product line will likely be better for meeting long-term goals, but a mined product line may provide a shorter time to market if the at-hand components are rich enough. Our participants included representatives of both approaches.
- **Long product life cycles.** This factor describes the average age of a product. Some domains, such as military avionics or certain types of information management, have notoriously long life cycles, as many as 20 years or more, while others have a lifespan of a year or so. This has implications for how often the product line will need to "turn over" in order to produce products that are markedly different from the original products to respond to the changing demands of their environment.
- **Product line maturity.** This factor describes how long the organization has been building product lines. A related issue is the **number of products** in the product line, which is a measure of the variability of the products. Together these suggest the breadth and depth of the organization's product line experience.
- **Criticality of adopting product line approach.** Half of our participants volunteered (without prompting) that the product line approach was not just a matter of convenience, creative engineering, or an idea that sounded good. It was, rather, a crucial step that their organization needed to take in order to survive. Those statements are paraphrased in the table.

These and other contextual factors—whether the software is built **internally or externally** to the organization that sells it, who each organization's **customer** is, and the organization's previous **domain experience**—are summarized in Table 2-1.

There may be many other factors in any category; these are simply the factors that surfaced during the presentations. A blank in the table means that the factor was not raised by that participant.

2-1. Contextual Factors for Product Lines Identified by Workshop Presentations

Contextual Factor	Participant 1	Participant 2	Participant 3	Participant 4	Participant 5	Participant 6
Assets designed or mined?	mined	designed	designed	designed	mined	designed
Long product life cycles?	no	yes	yes	yes		yes
Product line maturity	low	low: in demo stage	high	high	components beginning to be reused	high
Number of products	1-40	few	~12 PLs, ~65 applications	~60	21 components[a]	1-200 1-400

2-1. Contextual Factors for Product Lines Identified by Workshop Presentations

Contextual Factor	Participant 1	Participant 2	Participant 3	Participant 4	Participant 5	Participant 6
Criticality of adopting product line approach			"Couldn't afford to do business old way"	"Had to do something; else we'd be dead."		"Developing/maintaining separate products is path to disaster"
Software developed...	internally	internally and by contractor	internally	internally	internally	internally
Customer	external	parent organization	parent organization	external	internal "partners"	external
Previous domain experience	moderate		extensive, but electronics aspect is new	extensive	moderate and in-place	2nd generation

a. This participant represented a group that supplied reusable components to customers within the parent organization. Thus, he spoke in terms of the components, rather than the product lines in which those components were used.

In summary, we observed that the context for an organization fielding or beginning to field a product line can vary widely. Our participants represented organizations of different sizes, building vastly different products, and having different business goals. System size is not a prohibitive factor, as one product line comprises products ranging up to 1.5 million source lines of Ada. Neither complexity nor demanding requirements are prohibitive factors: two participants' organizations build hard-real-time embedded systems. Long product life cycles tend to be the norm, if not the rule. Extensive domain experience also tends to be the norm and (we suspect) the rule for successful product line development. Finally, most of our participants reported that migration to the product line approach was a matter of corporate necessity.

2.2 Technology Factors

Product lines involve technological support: What tool environments are necessary, or just plain useful? How are the commonalities and variations inherent in the different products captured? What role does architecture play? Key areas include the following:

- **Domain model artifacts.** The products in a product line populate a domain, and represent a set of variations on a common theme. Domain analysis is the name given to the structured process of understanding a domain in terms of its commonalities and variations. Whether this formal process is followed or not, a product line reflects the organization's vision of useful variability. This factor catalogs how our participants record that knowledge. A related theme is whether or not the organization adopted the **dual life-cycle model** in which domain engineering is the process used to create artifacts useful across the entire product line, and application engineering is the process used to produce a single product by adapting the domain-wide assets.

- **Tool support.** This factor records the tooling issues that arose during the discussion. The consensus was that better tools were needed. Three participants recorded a generative approach to products, meaning that they employed a tool that when given some sort of specification for a product would handle or assist in generating the code for that product.
- **Use of architectural concepts.** One path to managing a product line is to use a strong architecture blueprint to embody the commonalities across all the products, thus adopting a component-based plug-in approach to handle the variations. This factor described our participants' use of architecture as a unifying concept in developing and maintaining their product lines.
- **Importance of quality attributes.** Software in a product line must exhibit qualities above and beyond producing the correct answer, and even above and beyond those qualities that each individual product must exhibit, such as high performance or reliability. It must, for example, be easily produced and tested and adhere to a common architecture. This issue records how much attention was paid to the achievement of these special quality attributes by each organization, and when.

Finally, several participants mentioned that keeping pace with **new technology** posed a significant challenge all its own.

2-2. Technology Factors for Product Lines Identified by Presentations

Technology Factor	Participant 1	Participant 2	Participant 3	Participant 4	Participant 5	Participant 6
Domain model artifacts	Use cases, glossary	RLF[a] ODM[b], commonalities and variations		Analysis of customer + internal requirements	Requirements	"roadmaps" and specs; QFD
Dual life-cycle model?	yes	yes		implicit	no	yes
Tool support	Need some	GenVoca[c], COE[d], ATA[e], generative	Builder tool, libraries, generative. Problem: library management, QA	Rational Ada environment; need tools for non-reusable parts	many platforms	component assemblers; application configurator
Use of architectural concepts	object-based design[f]	layered, module design; extensive use of standards; COE	modular software design	layered, object-based design; strong application conventions	modular software, but white box reuse	"building blocks;" strictly layered, with parameterization. Architecture plays central role.
Importance of quality attributes		affordability was key	considered early: high reliability, serviceability	considered early		early: architecture is vehicle for achieving

2-2. Technology Factors for Product Lines Identified by Presentations

Technology Factor	Participant 1	Participant 2	Participant 3	Participant 4	Participant 5	Participant 6
Problems managing new technology?	yes	"keeping pace" is challenge	yes	yes: Ada, networking, compiler bugs,...	only as part of consuming projects	yes
Other technology factors				Insulation from new technology "religions;" parameter mgt.	finding needed info. on component	conformance to architecture; no description of executable architecture; hard to manage variation

a. RLF is Reuse Library Framework [Wallnau 88].

b. ODM is Organization Domain Modeling [STARS 96].

c. GenVoca is an application generator from the University of Texas at Austin [Batory 92].

d. COE is the Common Operating Environment, a Department of Defense standard data processing environment. See http://www.spider.osfl.disa.mil/dii/index.html.

e. ATA is the Army Technical Architecture, a COE-like standard for real-time embedded tactical systems. See http://www.hqda.army.mil/webs/techarch/homepage.htm.

f. Object-based design means that the software is divided into objects, but there is no inheritance.

In summary, an acceptance, even if implicit, of the life-cycle model that differentiates between activities of domain engineering and activities of application engineering was a common theme. However, the actual domain model artifacts and methods varied. What did not vary was the overriding importance of architecture to guide the design of the product lines. Product lines represent large-scale reuse of assets across products, but this is *planned* or *top-down* reuse as opposed to opportunistic or bottom-up reuse, which has proven unsuccessful in the field. Architecture is the carrier of the reuse plan, and is a valuable and reusable core asset in its own right. Notably absent from the discussion was the use of technology developed specifically for product line development.

2.3 Organizational Factors

Organizational factors include people and process problems that must be overcome before successfully fielding a product line. Key areas include the following:

- **Staff skills and attitudes.** Managing attitudes and engendering cooperation in the midst of a new way of doing business was an important factor. Intensive training (dozens of days per year) was the solution of one participant. One particular attitude that had to be overcome was the **one-at-a-time mentality** of building a system for its own sake rather than as a contributing effort to the organization's strategic goal of fielding a product line and building up a base set of core assets. Another attitude dealt with the preference of performing **domain engineering versus application engineering.** One participant reported that domain engineering was considered to be the glamour job in his organization, while building individual products was perceived as being less creative. Another participant, however, reported that in his organization application engineering was perceived as the most practical activity that yielded the only truly tangible results.

- **Responsibility for the architecture.** This factor captures who was responsible for the overall architecture for the product line. In most cases, the architecture was the purview of a small team of experts. Responsibility for the architecture means deciding what components each product will contain, how those components will interact with each other, and how those components will be tailorable to adapt to the particular needs of each individual product in the product line.
- **Terminology.** To our surprise, most participants reported that certain product-line-related buzzwords were taboo in their organizations because the words had lost all meaning or simply engendered confusion.
- **Management support.** Building a product line is not just an engineering agenda, it precipitates changes in personnel, personnel management, incentives, customer interfaces, scheduling, budgeting, and a whole host of management practices. It is a new way of doing business, and the consensus of our group was that if management does not vigorously and actively support the transition, the effort will fail.

2-3. Organizational Factors for Product Lines Identified by Presentations

Organizational Factors	Participant 1	Participant 2	Participant 3	Participant 4	Participant 5	Participant 6
Staff skills and attitudes	Recognize staff's level to carefully manage technology introduction		Developers questioned need for libraries; change, attitude, trust	Intensive training was key to establishing skills	Dedicated team; "high caliber people"	Roles were established to manage skills
One-at-a-time mentality	Was a problem		"But my system is unique!"	In beginning, project mgrs. competed	"must avoid not-invented-here syndrome"	not-invented-here syndrome a problem
Domain engineering vs. appn. engineering	Domain engineering more popular	Application engineering more practical				
Responsibility for the architecture	2 domain experts + 2 methods experts	working group of component vendor representatives	individuals	small team	owner + partners + baseline administrator	2 domain experts + 2 architects
Terminology: Don't use...		"reuse"	"reuse," "modular"	"architecture"		"components"
Management support		technical managers hard to convince	crucial	crucial	crucial	yes
Other management issues			Inconsistent processes			multi-site development

In summary, organizational and, in particular, people issues seem to play a paramount role in successful development of a product line. The concept must have strong support from above (management responsible for allocating sufficient resources to make the concept succeed) and from below (the staff charged with making it work). Overcoming mindsets from the previous way of doing business is difficult. In keeping with the importance of architecture mentioned in Section 2.2, an individual or a small highly-specialized team should be charged with the creation and caretaking of that architecture.

2.4 Business Factors

Whereas organizational factors address concerns within an organization, business factors address how the organization manages its external interfaces in the marketplace, and the benefits and risks to its position that are incurred when taking on a product line strategy. Key areas include the following:

- **Organization.** Most of our participants represented organizations that are arranged in business units of some form or another. A business unit is one that engages a particular market segment or (in the case of the U. S. Army) carries out a particular strategic mission. The product line serves them all, and must account for the different markets and missions of the business units in its variability.
- **Up-front investment.** By and large, our participants reported heavy up-front investment in adopting the product line approach. The investment was measured in dollars spent, as well as opportunity cost when production was suspended for a long time period while the first products were being built.
- **Engaging the market.** An organization that features a product line has economies and flexibility advantages that can be used to situate itself well in the marketplace. This area describes how the organizations represented at our workshop engaged their markets to make these advantages known and engender support. The case of participant #2 is somewhat unique: His organization's goal is to build a product line by having the suppliers of its components jointly agree on an architecture. This so-called trade association is meant to both define the architecture and engender participation by the leading vendors in that domain. Participant #4 reported the formation of a user's group in which his organization's customers meet and decide jointly on future requirements for the family of systems that they all share.
- **External interfaces.** This area describes issues with product lines from the point of view of stakeholders outside the development group. The sales people must know how to sell the product line to take advantage of their economics and flexibilities; intellectual property rights must remain with the developer rather than go out with each product, so that new products based on old ones can continue to be sold; and the schedules and needs of business units that consume the individual products must be considered.
- **Benefits.** This area describes how the benefits of a product line approach are measured by each organization that participated. Some measure the benefit in dollars as a return on investment. Most measure it, if only informally, by improved product quality and shorter development time. Some measure it in terms of increased staff productivity.

- **Risks.** The largest risks that were identified (besides the risks implicit in earlier issues such as staff attitudes) had to do with third-party suppliers, particularly COTS vendors. Half of our participants felt hostage, to some extent, to the whims of their commercial product suppliers. However, clearly this risk is not unique to product line development, but is borne by any developer of evolving systems.

2-4. Business Factors for Product Lines Identified by Presentations

Business Factor	Participant 1	Participant 2	Participant 3	Participant 4	Participant 5	Participant 6
Organization	business units	Army commands	business units plus "feeder group"	technical steering group; product groups plus PL group	business units; CSL is "feeder group"	lines of business
Up-front investment	big	big	no products were fielded	huge[a]	small	big
Engaging the market		trade association		user group	owner-partner model	
External interfaces		"must leverage the marketplace;" who owns the architecture? contractual incentives; architecture must be "competable[b]"; DoD policies may get in the way		intellectual property rights; educate customers; "our sales people understand"	process staged to fix 6-month development cycle	
Benefits	quality; education of staff	cost/savings (ROI in $)	productivity: better quality; quicker development	time to market; proven design; reliability; upgradeability; 80% reuse	large products done in record time; common look & feel; lower development/maintenance cost; engineers portable	high quality; increased productivity
Risks		Susceptibility to COTS vendors		Susceptibility to COTS vendors	Plethora of platforms; staying up on 3rd party items	

a. The participant asked that the figure be kept confidential.

b. "Competable" means that outside vendors must be able to compete fairly and equitably to supply parts (components) of the architecture.

In summary, a sobering commonality is that a significant up-front investment in developing core product line assets is normally required. The investment may be measured in dollars spent in training the staff or building the assets, or it may be measured in lost business because the organization output declines during product line inception. None of our participants indicated that they regretted the decision, but in some cases the decision brought with it a fair amount of pain.

3 Product Line Issues: Working Group Reports

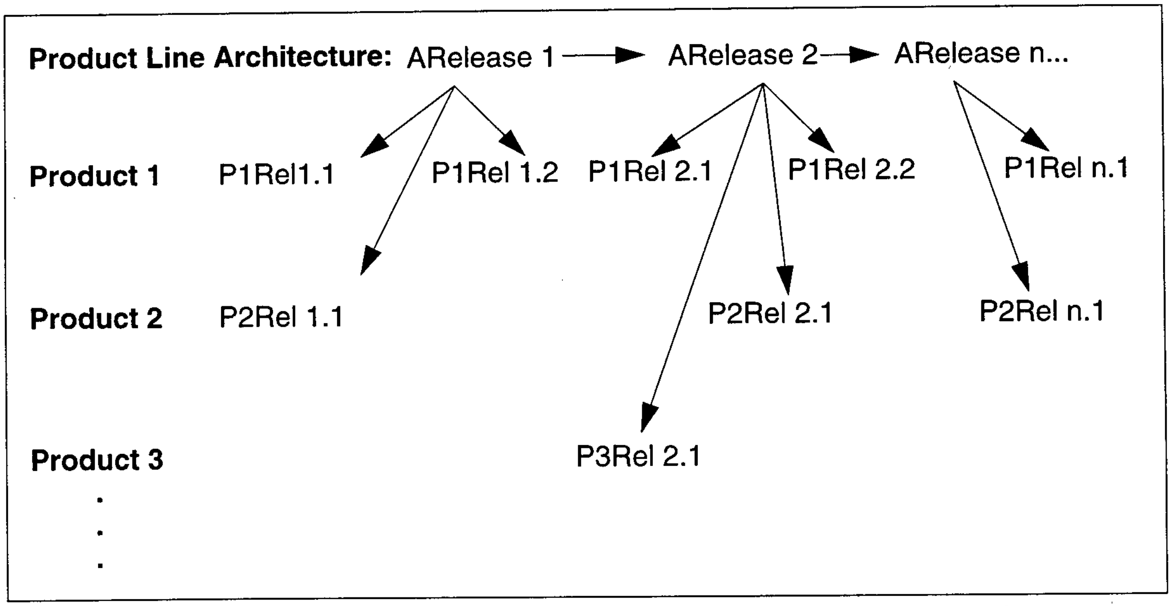
The issues that workshop participants felt the need to explore further fell into three categories: architecture and core assets, people-organization-management, and business models. Small working groups were formed to discuss each category. The reports of these smaller groups are found in the following sections.

3.1 Architecture and Core Assets

The architecture group discussed four major topics:

- How is a product line architecture and core asset set created and evolved?
- How are products kept synchronized with new releases of the product line architecture?
- How are products verified as being compliant with the product line architecture?
- What is the role of tools in the product line architecture support/dissemination process?

The basic evolution of a product line can be seen from Figure 3-1. In this figure, each architecture release (labeled ARelease i) provides the structure for potentially multiple releases of potentially multiple systems within the product line (labeled PiRel n.m) The product line architecture can include not only a plan for how the components of the systems are connected to and interact with each other, but the reusable components themselves, as well as other core assets (e.g., test plans) that will be used across products. In the figure, an arrow means "leads to."



3-1. Evolution of a Product Line Architecture and its Derived Products

We now treat the discussion issues in more detail.

3.1.1 Product Line Architecture and Core Asset Evolution

The length of time it takes to develop the initial product line architecture and deliver the first instance of the architecture was mentioned by at least two representatives as a serious risk in the adoption of product lines. Those attendees who had implemented product lines designed the first release of the product line architecture using experts in their particular type of systems. These experts accounted for domain knowledge, the software qualities they wished the product line to exhibit, knowledge of past systems in the domain, and knowledge of past systems and existing components that were to be incorporated into the product line.

A product line architecture evolves from its initial design and the initial release of the tailorable components and other core assets. This evolution results from the same factors causing the evolution of any system: desire to incorporate technological change, repair of existing problems, addition of new functionality, or restructuring of existing functionality to allow for more variants.

How are changes identified and then carried out? One of our participants reported a special group chartered to maintain the product line architecture and core assets. This group monitors both technology changes and individual product creation. Since only a small number of products are being created at any one time and since engineers are moved between the special product line group and the individual products, the maintenance of in-depth knowledge of both the product line and the products is accomplished.

In an organization with a large number of products under development or in the field, a more formal procedure must be used to identify and prioritize changes to the architecture and other core assets. This process, again, will likely mimic the process of selecting modifications to be made to any large system.

One participant reported a problem in evolving the core assets because of a corporate desire to preserve backwards compatibility. That is, when modifications are made to the core assets to produce a new product, it should be the case that all previous products should still be derivable from the core assets in their new form. "Derivable" means that the products should be buildable using whatever mechanisms, such as parmaterized instantiation, that were used to build them originally. Thus, there is at all times a single version of the core assets from which all fielded products are derived. This is a special case of the synchronization problem discussed in the next sub-section. In general, the problem is not so much a technical problem of updating the product line as it is a management problem of allocating resources to maintain the correspondence between core product line assets and fielded products.

Sometimes management seems to be interested in the initial creation of the product line, but then loses interest in its evolution. This causes upgrading of the product line architecture and core assets, as well as previously-developed products, to be given lower priority; hence, products "drift" away from the product line architecture, resulting in increased maintenance cost and lower responsiveness. A product line architecture and core assets evolution process must guard against such drift.

3.1.2 Synchronizing Products with New Releases of the Product Line Architecture

The problem of synchronization occurs when developing a new release of a product. Referring back to Figure 3-1, consider the development of release 2.1 of product 1. Release 1.1 of product 1 was based on release 1 of the product line architecture. Since that time, release 2 of the product line architecture has been made available.

Suppose a new release of a product is required. Suppose further that since its initial release, a new version of the core assets has become available. It would maintain the integrity of the product line to base the new product release on the current version of the core assets—but it may very well be cheaper to base it on the version of the core assets that begat it in the first place. Thus, over time, the product will diverge from the core assets, or multiple versions of the core assets will have to be maintained.

In many ways, this problem is similar to the problem of basing a particular system on a COTS product. The COTS vendor may issue new releases from time to time and a vexing question for the system developer is whether, how, and when to incorporate those releases into the system. In a product line environment, features could potentially be incorporated into the product line to facilitate the synchronization of the product line and the products.

One participant reported that their process finesses this issue. When they create a new product from the product line assets, they create it from the latest available version of those assets. They then make no effort to keep that product current with evolutions in the product line assets. Instead, from time to time, they change the product line assets to incorporate modifications suggested by the creation of the product.

Keeping future releases of a product in synchronization with the product line core assets is essential to prevent product line degradation. If the core assets are not updated to reflect the latest products, variants will be created from products rather than from the product line assets and, over time, components will not be reusable and the benefits will be lost.

3.1.3 Conformance of Systems with the Product Line

During construction, systems tend to drift from the original architecture and the original design. This is true even in one-of-a-kind systems, in which the only requirement associated with this drift is to ensure that the as-built system is documented. In a product line, the consequences are potentially more serious. If the drift is too severe and too early then components that are a portion of the product line cannot be used without modification. A more likely concern is that components created for a single system must have a great deal of work performed on them before they are available for inclusion in the product line.

It is not clear that this is a problem in practice. Perhaps it is obviated when architecture takes a strong role in successful product line development. Nevertheless, "system drift" is clearly a potential danger, and a disciplined development process is required to keep it in check.

3.1.4 Tool Support

Our participants reported very little in the way of technological support specifically aimed at fielding a product line, as opposed to tools generally helpful in producing one-of-a-kind systems. On one hand, the existence of product lines without special tool support indicates that tools are not a necessary part of the development of product lines. On the other hand, a number of places in the product line process were identified as places where tool support could be very useful.

One of the participants uses a builder tool to deliver individual products. The tool works by asking its user a number of questions about the desired characteristics of the product being built. It then tailors the components automatically and delivers an assembled version of the system that meets those characteristics. This is likely a large contributor to the success of their product line effort. It enables any member of the organization who needs to produce a product—a customer service representative, for instance—to do so. Thus, the product line concept is made accessible throughout the organization.

Since one of the largest risks in adopting a product line strategy is the length of time before the first instance is delivered, tools and techniques to support timely initial delivery are important. Given the need for domain experts to design the core assets, the use of tools that experts in a domain can easily use to capture designs, design alternatives, and design rationale is an advantage. If conformance of individual products to the product line core assets turns out to be a problem, then techniques to validate conformance should be identified and tools should be adopted to support these techniques.

Once the strategies for ensuring synchronization are understood, tool support will be useful for moving between individual products and the product line core assets.

Tools are intended to expedite certain processes. A decision as to which tools are appropriate to establish and support a product line depends partially on an understanding of which processes are troublesome and costly. Architectural, business, and process issues all apply to this problem, which provides a ripe area for future work.

3.1.5 Summary

Architecture is usually the basis for product lines because it embodies the earliest design decisions for the product line and provides a framework within which reusable components can be developed. Some problems in constructing a product line can be couched in terms of the evolution and representation of the architecture. By couching the problems in this form, issues of synchronization and initial design become apparent. Other problems in the process such as providing a business case for adopting the product line approach or for educating the rest of the organization about the product line may also be more clearly understood in an architectural context. Any organization that embraces a product line approach must identify practices that clearly address asset evolution, synchronization, and appropriate tool support. Product drift is a pitfall that, unless carefully monitored, will prevent success of product line development.

3.2 People-Organization-Management

Presentations and discussions at the workshop demonstrated that the technical issues, while being addressed, have not been solved. In some ways, we are facing a situation not unlike the functional/non-functional system requirements problem: In designing a system, we must examine the functional requirements in light of the non-functional, such as maintainability, security, and performance. For product lines, advancing the technical issues must be done in light of the organizational, non-technical product line requirements.

These non-technical requirements include the following:

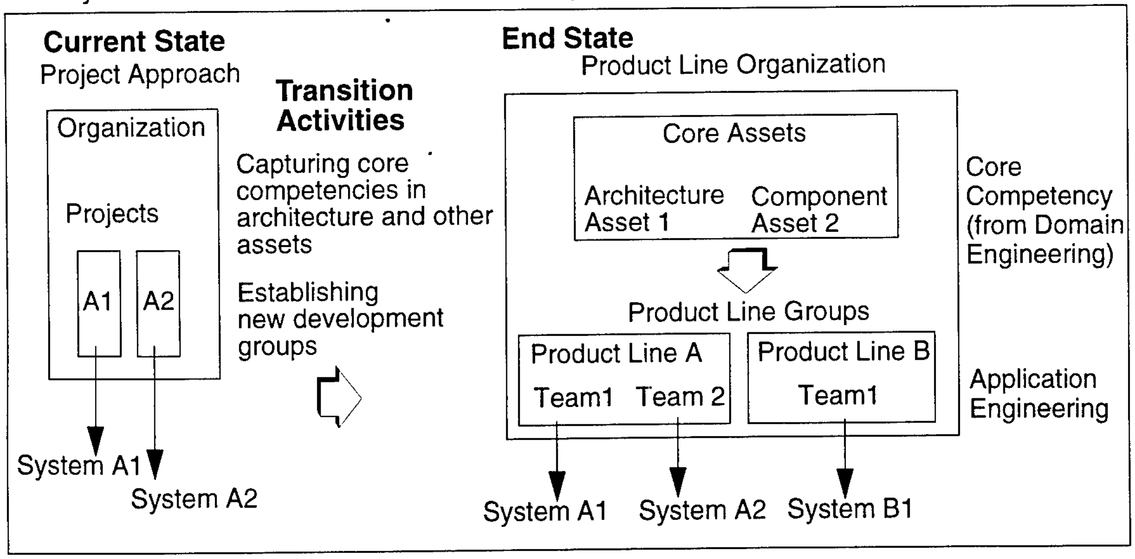
- How does an organization change as product lines become institutionalized?
- Who are the participants in a product line organization?
- What are their roles and responsibilities in terms of requisite skills, required training, and cultural outlook?

The following subsections address these requirements in turn and then discuss the open issues identified by the working group.

3.2.1 Transitioning to a Product Line Organization

There is a direct contrast between the project approach to developing systems and the product line approach. Most organizations today focus on individual systems. These organizations usually derive new systems from previous instances. They may organize these systems into product lines, but achieve only minimal code sharing and design concept reuse. The most common approach to deriving a new system involves utilizing the personnel who developed previous systems. Many organizations realize that this path, while expedient, is labor-intensive and cannot be sustained as they move to a highly competitive marketplace, where time-to-market, market share, and customer satisfaction are drivers. Sufficient numbers of experienced personnel are not available to staff each project within the product line to support the growing demands for new software systems. An alternative must be found that captures the knowledge of these experts and allows it to be leveraged across new systems.

One alternative approach focuses on product lines that derive new systems from a core architecture and components. Development of these core assets comes from corporate investment in product line R&D that captures system information and the competencies of the product line experts. Corporate use of these core assets is guided by rules establishing organizational dependencies, which are monitored at the corporate level. Figure 3-2 is based on the experience of the working group members and shows the transition from project to product line orientation. The core assets in the figure are shared between the two product lines. Other assets are unique to the individual product lines. In other product line organizations, the individual product lines may have their own architecture and components and may do little sharing of assets.



3-2. Transition to the Product Line Organization

As an organization makes this transition it will typically go through several stages:

Stage 1. In the early stage, the organization looks to systems experts and legacy systems to contribute to the core understanding. It captures baseline measurements to assess its current state and analyzes its product mix to identify product line opportunities.

Stage 2. This may be called the "Black Hole Stage." The organizational structure is formed to develop core assets. During this stage there is heavy investment in assets, but low product yield. The organization should measure investment levels for core assets and forecast investment return.

Stage 3. During this stage, the product line groups begin to deliver systems. The core asset group also evolves and maintains the core assets. Here, the organization can evaluate return on investment.

Stage 4. In the final stage, the product line organization has been institutionalized. The R&D group now focuses on identifying new areas for applying the product line approach and achieving economies of scope by applying existing assets to new product lines.

This progression does not occur in strict linear fashion. There is a cycle as product lines mature, are retired, and new product lines emerge.

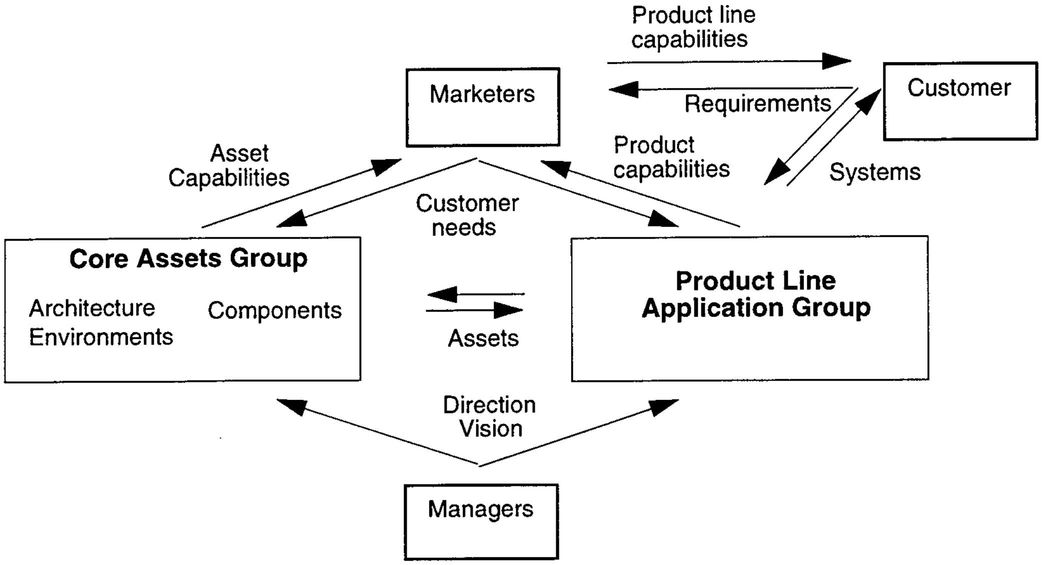
3.2.2 Key Players in the Product Line Organization

The working group identified the key players in a product line organization:

- Customers - purchasers of a system; may be end users or may represent needs of a group of users
- Marketers - relate product line capabilities to prospective customers; relate customer needs to asset and application developers

- Core assets group - develops architecture and other assets for product line
- Application group - delivers systems to customer
- Managers - provide and sustain product line vision

Figure 3-3 shows these organizations and some of the key interrelationships.



3-3. Relationship of Players in Product Line Organization

3.2.3 Organizational Roles and Responsibilities

Each of the roles identified within the product line organization has a required set of skills and requisite knowledge relating to product lines and product line development. The skills and knowledge may come from training or from mentoring within the organization. The cultural values held by the product line organization are also key to the success of the product line approach. Table 3-1 lists the roles and their respective training and also gives a sense of the culture that must exist within the product line organization.

3-1. Roles and Responsibilities in Product Line Organizations

Role	Skills	Requisite Training and mentoring	Culture
Marketers	• domain understanding • commonality needs and trends • understanding of customer needs • understanding of trends • negotiating • salesmanship	• features of product line • examples of delivery, cost, performance of product lines	• sell product line; not people • do communicate needs and trends to core asset team

3-1. Roles and Responsibilities in Product Line Organizations

Role	Skills	Requisite Training and mentoring	Culture
Customers	• communicate needs • negotiating • communicate trends	• benefits of product line • vocabulary of product line • overview of product line principles and architecture	• confidence in asset investment • confidence in products in product line
Core Assets Group	• mediating • domain and application understanding • technically current • technical lead/chief designer levels • abstraction skills (meta-design)	• application technology • application domain • designing for reuse • product line principles and processes	• receptive to marketing and customer input • satisfaction in designing general pieces (Culture lines of distinction with application builders blur where team concept is mature)
Application Group	• ability to engineer from "building blocks" • application understanding • ability to interface with customers • technical expertise to implement systems	• application technology (languages, tools, etc.) • problem domain • product line principles and processes (Training levels will depend on organizational product line maturity)	• satisfaction in delivery of systems • supports building block approach • uses building blocks over "growing own" • team-based
Managers	• authority • vision • technically savvy • leadership	• product line development process • business model	• must have training plan • must reward product line practice • confidence in approach and other players • supporting investment for long-term pay back

3.2.4 Success Factors

In summary, the working group identified several key issues in supporting the transition and sustainment of the product line organization:

- Qualification for position versus on-the-job training

  Table 3-1 identifies the players in the product line organization and states their skills and training requirements. While some of this training may be formal, much will come on the job, through monitoring or osmosis. Management must be aware of this need and identify opportunities for new staff to obtain the required knowledge; management must also hire staff who are adaptable to this new approach.

- Interface with individual customers ("partners")

Under the product line arrangement, there will be a close relationship between customers and the product line development groups. This was reflected in the fact that one of the participant's organizations now calls their customers "partners." The customer will negotiate his/her requirements in light of product line capabilities. Marketing must be able to relate these capabilities to customers and describe the benefits of the product line and the product line approach. It is the responsibility of the development groups (core asset and application) to provide information to marketing allowing them to create and sustain these new customer relationships.

- Interface with business units (aggregate of customers)

  Management of the product line organization must establish ties to other business units. Over time, the product line approach should become the standard way of doing business for an entire organization. To achieve all the benefits of product lines, business units must coordinate their activities, partitioning and scoping product lines in light of customer needs. An organization must make sure that the entire marketing organization -cutting across business units - is aware of these partitionings and can relate to customers, matching them to the appropriate product lines, seeing gaps that existing product lines may fill, and identifying new areas of work.

3.3 Business Models

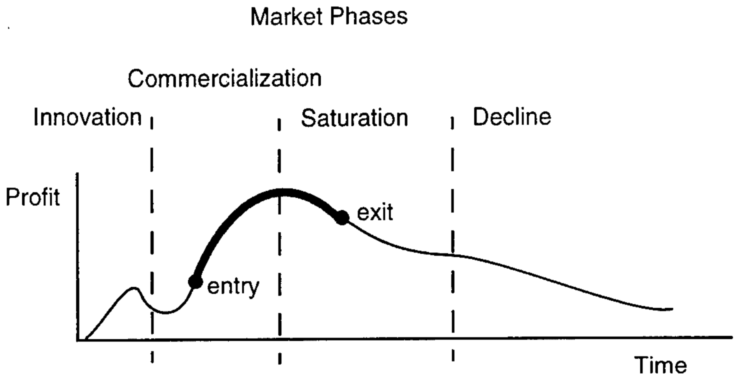
The business model group chose to explore the business conditions for which a product line approach to software engineering can be recommended. To support decisions regarding the timing and implementation of a product line approach, the group drafted a couple of analytical models. While further work is needed to complete these models, they begin to address some of the following business issues:

- Under what business conditions is a product line approach to software engineering an appropriate competitive response? What can be said about market conditions, product lifecycles, competition, intellectual property, and other business factors that would cause a decision maker to recommend a product line approach?
- What is the market timing for implementing a product line approach? It takes time to implement a production system. Technology and markets change, increasing the risk that product line assets may fail to provide any competitive advantage before their investment expense is recovered.
- What *product line production strategy* should be adopted given market and competitive forces? Depending on the business goals, the software technology, the customers and the competition, the differences in products (i.e., the content of variation), and the scope of the products (i.e., the range of variation) supported by a product line approach will vary.

This in turn affects the choice of assets, processes, staffing, and functions of the production system.[1] A framework that relates the different forms of a product line approach to an organization's business environment is needed. What business factors influence the choice of a product line approach?

3.3.1 Timing the Introduction of a Product Line Approach

To understand market timing issues regarding the adoption of a product line approach, the working group modeled a typical product/profit life cycle. The modified "S" curve in the figure below shows the anticipated profit for a line of products over four market phases: *Innovation, Commercialization, Saturation*, and *Decline*.



3-4. A Typical Product/Profit Lifecycle

Beginning with the *Innovation* phase, the product is new, perhaps a first offering from a company launched on venture capital. The market is untested: The company is learning about their customers and the product features for which they are willing to pay a premium. The *Commercialization* phase begins when a generic category has been established in the minds of a group of customers, and the features embodied in the new product fit that category. (That is why profits often dip; getting name recognition and establishing a customer base is often expensive and fortuitous). Once the market is established, many companies will enter the market, and *time-based competition* becomes important. (See the market entry point in the figure.) Later the market becomes saturated with choices—the beginning of the *Saturation* phase. Features for different market segments become stable; faster time-to-market of new product line products does not appreciably increase market share or profit; companies begin to com-

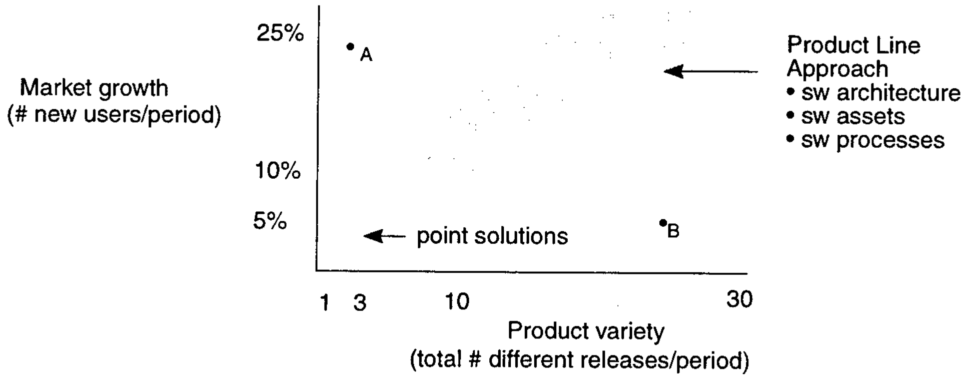
[1] Presentations given at the workshop described different production strategies that had been implemented. One organization focused on cost reduction, developing an architecture and large-grain common subsystems, achieving up to 80% reuse. Another organization focused on automation generation of software code so that 1) domain experts in the field rather than software engineers could generate application software, and 2) products would be more quickly available after the hardware design had been completed. Another organization focused on the interoperability of the software provided to a service organization, and developed assets that provided low-level common functionality and a common look and feel to end users.

pete on price. A shake-out occurs: Companies start exiting the market (see the market exit point in the figure), and the mix of products shrinks to a core set. Finally in the *Decline* phase, the products become commodities; there is little discrimination on features, and profits fall to a low but stable point.

Thus, as indicated by the thick section of the curve, a product line approach is best implemented during the upswing of the commercialization phase and continued through the middle of the saturation phase of the product/profit lifecycle. Planning for a product line approach (including an investment analysis) should be conducted at the beginning of the commercialization phase so an organization is prepared should the market emerge.

Timing can be monitored by measuring the extent to which market growth is determined by the rate at which new products are introduced. When growth is positively related to product differentiation (i.e., new releases have features that create new groups of customers), a product line approach is feasible. This is shown as the shaded region in the following figure [Withey 96].

The approach may not be feasible at points A or B. At point A, market growth is not driven by the introduction of a variety of products. Product sales are high for some other reason, most likely because the product incorporates new technology that provides a benefit or a service not yet offered by the competition. At point B, the market is saturated with choices. Additional products do not contribute greatly to growth or income. Few changes are needed, and unless development costs are high, streamlining production costs through software assets may only provide small return on investment [Withey 96].



3-5. Conditions for Product Line Approach

3.3.2 Production Strategies by Product/Profit Lifecycle

Obbink describes a taxonomy of product line production strategies [Obbink 95] that maps well to the different market phases of the product/profit lifecycle. With more elaboration, this taxonomy could help managers determine the most effective form for a product line approach given market conditions. In the following table the different production strategies are distinguished by the intended variability in the product line, the nature of the architecture and assets, and the software modification process.

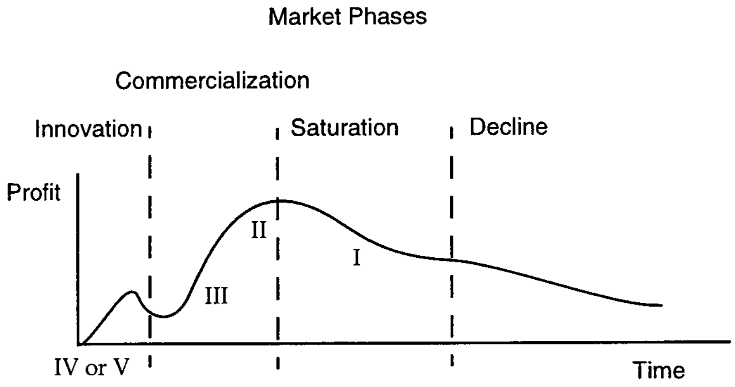
Production Strategy	Description	Market Phase
I	Variation: restricted to changes in user interface and selection of already implemented functionality Architecture and Assets: stable architecture and automatic code generation Product engineering process: concentrates on modification of user interfaces and the integration of function embodied in software assets	Saturation Features for different market segments become stable. The rate of product introduction has slowed, and the mix of products in the line shrinks to a core set. Companies begin to compete on price.
II	Variation: new functionality that fits within design constraints of architecture can be added Architecture and Assets: robust, evolvable architecture and components Product engineering process: concentrates on architecture evolution and the development or adaptation of software assets to incorporate new functionality. For unaffected components, production strategy I is followed.	Late Commercialization The market is established. Many companies are entering the market, and the introduction of new features faster than the competition becomes important.
III	Variation: new technology can be incorporated Architecture and Assets: architecture describing problem space is stable, new implementation architecture and components Product engineering process: full-fledged domain engineering including the design and implementation of production strategies I and II	Early Commercialization A generic product category has been established in the minds of a group of customers, and the features embodied in the new product fit that category. A product line is likely.

3-2. Production Strategies

Production Strategy	Description	Market Phase
IV	Variation: entry into new domains such as multimedia Architecture and Assets: unstable Product engineering process: concentrates on research and development to integrate domain into existing domains	Innovation The product is new, a product line in not yet established. The market is untested: The company is learning about the technology, their customers, and the product features for which they are willing to pay a premium.
V	Variation: one-offs: first-of-a kind products; variation in domain knowledge, functionality and technology Architecture and Assets: none Product engineering process: concentrates on maturing technology and functionality, small group, start-up	

3-2. Production Strategies

The working group mapped these strategies to the "S" profit curve modeled earlier. As shown in the following figure, a new product will typically start as the result of a type V or IV process. When it has sufficient mass-market potential it will evolve to a type III process. Near the peak of the commercialization phase, a type II production process will be used. To survive a market shake-out, a type I process may be implemented.



3-6. Product Line Approaches by Market Phase

3.3.3 Business Drivers Affecting Product Line Approaches

The working group then began exploring the business drivers that determine the product line production strategy that is implemented.

The group considered two drivers: the size of the market and the ownership of the unique product knowledge incorporated in the product line. Regardless of market phase, a product line approach is not always desirable. The up-front investment expense may not be recovered if the market cannot sustain the break-even number of products or if the product knowledge embodied in the architecture and assets can be easily copied by the competition. Knowing the size of the market and the extent that product knowledge is and can remain proprietary can help managers make decisions about which product line strategy to pursue.

Product knowledge is either public or private. If private, the knowledge is held within the organization. The organization has considerable intellectual property protected by trade secrets, patents, and copyrights. Typically, the organization has made sizable investments; this private knowledge is usually the product of extensive technical and market research and development. The architecture, assets, and processes are invisible to the customer. If public, the knowledge is widely available. Much of the knowledge and expertise is available commercially from suppliers, universities, or consulting companies. The organization has invested little in developing its own knowledge base.

The market size can be described by the number of different groups of customers served by a line of products. A niche market is defined by the common needs/requirements of one group of customers. A mass market, on the other hand, can be segmented into many customer groups, with multiple products targeted to the specific needs of each group. A mass market may provide a sufficiently large revenue base to warrant investments in software architecture, assets, and processes.

Speculating on the implications of these dimensions, the group produced Table 3.3. A different production strategy for a software product line is described in each of the cells formed by these two business drivers. For each strategy, the factor that drives product differentiation is listed as well as a characterization of the processes, assets and software architecture that compose a product line approach.

A discussion of the cells of the table follows.

Companies in cell A develop products that incorporate new technology for a niche market. Because the market is small, they typically adopt a preemptive strategy using technology to beat competitors and to either transform the market or gain market share. Because technology often radically changes the implementation of the products in software, reusable code components and interfaces are not often feasible. However, since the market is established, a conceptual model of the primary functions provided by software systems in the product line can be developed. Since new technology is the source of competitive advantage, companies usually invest in prototyping and simulation tools. An example of a company in this cell is a missile developer for the Department of Defense.

Companies in cell B race against competitors to introduce new products that meet the needs of specific customer segments. A proprietary architecture that achieves high levels of component reuse and functional integration over a wide product mix is a key source of competitive

	A	**B**
Private Product Knowledge	differentiation: technology architecture: established architectural style, conceptual architecture, narrow scope process and assets: tool-based integration and simulation	differentiation: features architecture: established architectural style, well-defined multiple views, wide scope process and assets: tool and component based
	C	**D**
Public Product Knowledge	differentiation: custom products to order architecture: none process and assets: craft-based, labor intensive, assets vary by individual	differentiation: price and service architecture: public or commercial standard: protocols, interfaces, functions process and assets: variable
	Niche Market (Single segment)	**Mass Market (Multiple segments)**

3-3. Different Product Line Approaches by Two Business Drivers

advantage. Because the products interface poorly with a competitor, a customer often cannot buy from a competitor without duplicating the investment and adding complexity. Customers are essentially locked into purchasing from the company. To keep a company from dominating the market, competitors will often file anti-trust suits. Examples of companies in this cell are telecommunication switch developers.

Companies in cell C compete on the ability to provide custom solutions. Any technology that is developed is done on contract; intellectual property rights are not retained. They concentrate on only a few customers. Since companies typically take advantage of any business opportunity, the organization does not pursue the development of a core competency; an architecture for a family of products is not developed. Development processes and assets are highly individualized. Contract software development firms are examples of companies in this cell.

In cell D, customers build systems from commercial components that are compatible with a public standard or commercially-available architecture. Companies, therefore, compete largely on component price and convenience. They adopt a blocking strategy, using market strategies to dominate a niche and make competitive entry unattractive. Focusing on a niche market, they may develop a proprietary technology for a component or a set of components, and thus strategically move to cell A.

Given these two business factors, the above table gives some initial guidance on the issues and decisions to consider when implementing a product line approach. It shows that a product line approach must be integrated with an organization's business strategy.

The table also explains the differing, and somewhat conflicting, strategies of government and industry regarding a product line approach. To reduce the number of unique systems that essentially satisfy the same mission, and to reduce acquisition and maintenance costs, the government is attempting to move industry from cells A, B, and C to cell D. An example of this strategy is the Software Market-Driven Architecture Trade Association (SMART) Initiative pursued by the Army to establish application architectures and common assets. To grow the nation's technology base, the government, through subsidized research and development, also seeks to move companies from cell C to cell D. Because of high profit margins, industry, on the hand, tries to remain in cells A and B. If technical or market forces dictate, companies will collaborate to establish a standard architecture and make money in applications or components.

3.3.4 Issues and Further Areas of Investigation

The working group feels more work is needed on the following:

- A better understanding of the relationship between an organization's business environment and the elements of a software product line approach—the architecture, assets and processes. How do the product line production strategies relate to market cycle time?
- How an organization evolves a product line approach when assets or entire products are developed by software suppliers. How are an architecture and assets acquired in government programs?
- Cost/benefit models for different product line production strategies. To improve chances of a high return on investment, what are the prerequisites for a product line approach? What software assets should be developed?
- Risk mitigation strategies. How does an organization manage product development until a robust architecture and assets are available?

4 Summary

This report is the first in what we hope will evolve into a series of reports on forums at which product line practice is addressed. Based upon the workshop discussions, it is clear that there are several identifiable critical factors in product line success. These factors include deep domain expertise, a well-defined architecture that guides the design of the product line, a distinct architect or architecture team charged with the creation and caretaking of that architecture, a solid business case for the product line, visible management commitment and support, a distinct organizational entity responsible for maintaining the reusable assets, robust configuration control practices, and well-defined customer management. It is also clear from the discussion in the working groups that many technical and non-technical issues have not been resolved. Though our discussions on architecture, people-organization-management, and business models probed at the issues, many problems remain unsolved. Perhaps most importantly, the repeatable integration of technical, organizational, and business practices remains a challenge.

Consequently, there appears to be a need for considerably more exploration and codification of both technical and non-technical product line practices and for periodic forums for sharing non-proprietary ideas, techniques, and lessons learned. To that extent the SEI intends to continue holding similar workshops and will also continue to report the workshop results to the software development community at large.

We expect that the information in this report will be refined and revised as the technology matures and as we continue to receive feedback and to work with the growing community of software engineers championing a product line approach. If you have any comments on this report and/or are using a product line approach in the development of software-intensive systems and would like to participate in a future workshop please send electronic mail to lmn@sei.cmu.edu.

Glossary

application engineering

an engineering process that develops software products from partial solutions or knowledge embodied in software assets

business model

a framework that relates the different forms of a product line approach to an organization's business context and strategy

core asset

a software asset or other investment (such as training, estimates, work breakdown structure) that is used in multiple systems

domain

an area of knowledge or activity characterized by a set of concepts and terminology understood by practitioners in that area

domain analysis

process for capturing and representing information about applications in a domain, specifically common characteristics and reasons for variability

economies of scale

the condition where fewer inputs such as effort and time are needed to produce greater quantities of a single output

economies of scope

the condition where fewer inputs such as effort and time are needed to produce a greater variety of outputs

Greater business value is achieved by jointly producing different outputs. Producing each output independently fails to leverage commonalities that affect costs. Economies of scope occur when it is less costly to combine two or more products in one *production system* than to produce them separately.

investment analysis

a process of estimating the value of an investment proposal to an organization

Investment analysis involves quantifying the costs and benefits of the investment, analyzing the uncertainties, and constructing a spending strategy. This analysis links the strategic and technical merits of an investment to its financial results.

product family

a group of systems built from a common set of assets

product line

a group of products sharing a common, managed set of features that satisfy specific needs of a selected market or mission

product line approach

a system of software production that uses software assets to modify, assemble, instantiate, or generate a line of software products, i.e., building a software product line as a product family product line architecture

description of the structural properties for building a group of related systems (i.e., product line), typically the components and their interrelationships. The guidelines about the use of components must capture the means for handling variability discovered in the domain analysis or known to experts.

product line system

a member of a product line

production system

a system of people, functions and assets organized to produce, distribute, and improve a family of products. Two functions included in the system are domain engineering and application engineering.

software architecture

description of the structural properties of the software, typically the components and their interrelationships and guidelines about their use [Clements 96]

software asset

a description of a partial solution (such as a component or design document) or knowledge (such as a requirements database or test procedures) that engineers use to build or modify software products [Withey 96]

system architecture

software architecture plus a specification of execution and development environments

References

[Batory 92] Batory, D. & O'Malley, S. "The Design and Implementation of Hierarchical Software Systems with Reusable Components." *ACM Transactions on Software Engineering and Methodology 4*, 1 (October 1992): 355-398.

[Brownsword 96] Brownsword, Lisa & Clements, Paul. *A Case Study in Successful Product line Development* (CMU/SEI-96-TR-016; ADA315802). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1996.

[Clements 96] Clements, P. & Northrop, L. *Software Architecture: An Executive Overview* (CMU/SEI-96-TR-003; ADA305470). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1996.

[Moore-McKee 94] Moore-McKee, Amy L. "Application Builder: A Method for Building Application Software for Caterpillar Electronic Engines," 67-72. *Proceedings of the 16th Annual Fall Technical Conference on the Intercombusiton Engine Division ASME*. Lafayette, IN, October 1994. New York, NY: ASME, 1994.

[Obbink 95] Obbink, J. "Product Differentiation and Integration: the Key to Just-in-Time Product Development." *Lecture Notes in Computer Science* 913 (1995): 79.

[Rosenbaum 95] Rosenbaum, Susan & du Castel, Bertrand. "Managing Software Reuse—An Experience Report," 105-111. *Proceedings of the 1995 International Conference on Software Engineering*. Seattle, WA, April 23-30, 1995. New York, NY: Associating of Computing Machinery, 1995.

[STARS 96] STARS. *Software Technology for Adaptable Reliable Systems. Organization Domain Modeling (ODM) Guidebook*. Version 2.0. (STARS Technical Report STARS-VC-A025/001/00). Lockheed-Martin Tactical Defense Systems, Manassas, Virginia, June 1996.

[van der Linden 95] van der Linden, Frank J. & Müller, Jürgen K. "Creating Architectures with Building Blocks." *IEEE Software 12*, 6 (November 1995): 51-60.

[Wallnau 88] Wallnau, K., et al. "Construction of Knowledge-Based Components and Applications in Ada." 3/1-21. *Proceedings of the Fourth Annual Conference on Artificial Intelligence and Ada*. Fairfax, Virginia, Nov 15-16, 1988. Fairfax, VA: George Mason University, 1998.

[Withey 96] Withey, J. *Investment Analysis of Software Assets for Product Lines* (CMU/SEI-96-TR-010; ADA315653). Pittsburgh, PA: Software Engineering Institute, 1996.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. **AGENCY USE ONLY** (leave blank)

2. **REPORT DATE**
June 1997

3. **REPORT TYPE AND DATES COVERED**
Final

4. **TITLE AND SUBTITLE**
Product Line Practice Workshop Report

5. **FUNDING NUMBERS**
C — F19628-95-C-0003

6. **AUTHOR(S)**
Len Bass, Paul Clements, Sholom Cohen, Linda Northrop, James Withey

7. **PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

8. **PERFORMING ORGANIZATION REPORT NUMBER**
CMU/SEI-97-TR-003

9. **SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
HQ ESC/AXS
5 Eglin Street
Hanscom AFB, MA 01731-2116

10. **SPONSORING/MONITORING AGENCY REPORT NUMBER**
ESC-TR-97-003

11. **SUPPLEMENTARY NOTES**

12.a **DISTRIBUTION/AVAILABILITY STATEMENT**
Unclassified/Unlimited, DTIC, NTIS

12.b **DISTRIBUTION CODE**

13. **ABSTRACT** (maximum 200 words)

The first Software Engineering Institute Product Line Practice Workshop was a hands-on meeting held in December 1996 to share industry and government practices in software product lines and to explore the technical and non-technical issues involved. This report synthesizes the workshop presentations and discussions, which identified factors involved in product line practices and analyzed issues in the areas of architecture, people-organization-management, and business models.

14. **SUBJECT TERMS**
domain engineering, product family, product line practice, software architecture, software product lines

15. **NUMBER OF PAGES**
36

16. **PRICE CODE**

17. **SECURITY CLASSIFICATION OF REPORT**	18. **SECURITY CLASSIFICATION OF THIS PAGE**	19. **SECURITY CLASSIFICATION OF ABSTRACT**	20. **LIMITATION OF ABSTRACT**
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL